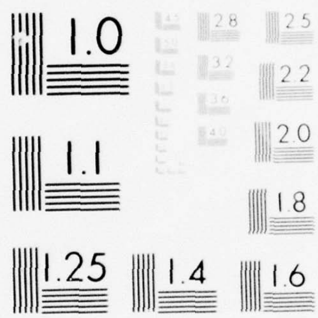
AD-A046 596 FRANKFORD ARSENAL PHILADELPHIA PA F/G 9/2
FUNCTIONAL DESCRIPTION OF THE OPERATIONAL PERFORMANCE ANALYSIS --ETC(U)
1975

UNCLASSIFIED FA-FCF-1-75 NL

| OF |

AD
A046596

END
DATE
FILMED
12-77
DDC


1.0
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

(1)

B. S.

(14) FA-
REPORT FCF-1-75

(11) 1975 (12) 21 p.

(6) FUNCTIONAL DESCRIPTION OF THE
OPERATIONAL PERFORMANCE ANALYSIS LANGUAGE (OPAL),

(9) Letter Rept.
1

Approved for public release; distribution unlimited.

Department of the Army
FRANKFORD ARSENAL
Philadelphia, PA 19137

142 72Ø

JB

FOREWORD

This Letter Report contains the functional description of the Operational Performance Analysis Language (OPAL). This specification of the language is based upon language constructs and concepts derived from the Abbreviated Test Language for Avionic Systems (ATLAS), developed by the ATE Subcommittee of the Airlines Electronic Engineering Committee; the Ground Operations Aerospace Language (GOAL), developed for the National Aeronautics and Space Administration; the Compass Test Language (CTL), developed by Massachusetts Computer Associates under contract to Frankford Arsenal; and the general purpose, high-level languages FORTRAN, ALGOL, and PL/1. It includes contributions developed by the AMC-ATE Language Standardization Committee during the past two years, as well as those developed under the Defense ATE Language Standardization (DATELS) project during over one year of effort.

Appendix I traces the derivation of each of the concepts and constructs specified.

Comments relative to this report should be forwarded, in writing, to:

Commander
Frankford Arsenal
ATTN: SARFA-FCF-C
Philadelphia, PA 19137

ERRATA SHEET--REPORT FCF-1-75
FUNCTIONAL DESCRIPTION OF THE OPERATIONAL
PERFORMANCE ANALYSIS LANGUAGE (OPAL)

Page 8:

1. Change first line of text under FORM 4 from:

"where the specified block is executed repeatedly until the specified condition..."

to:

"where the specified block is executed repeatedly while the specified condition..."

2. Change third line of text under NOTE (second paragraph top of Page 8) from:

"is met, or if the condition (FORM 4) is true,..."

to:

"is met, or if the condition (FORM 4) is not true,..."

Functional Description of the
Operational Performance Analysis Language
(OPAL)

1. INTRODUCTION

1.1 Maintenance operations are increasingly dependent on automatic test equipment (ATE) to minimize the time, cost, and critical skills required to test, diagnose, and maintain the widely varied inventory of Defense materiel. The major problem in the use of any automated system is the development of reliable computer programs to control the required operations. In the maintenance field, this problem can be solved only if means are provided to simplify the task of programming ATE. Essential to such simplification is the availability of a standard test language.

1.2 A master plan for the development of the standard language, to be called the Operational Performance Analysis Language (OPAL), and of its software support system, was published as part of the Defense Automatic Test Equipment Language Standardization (DATELS) effort.(1) Major objectives of the project were stated as follows:

1.2.1 Promote the development of a standard, high-level, programming language which will improve communications and understanding between test system developers, and users, and unit under test (UUT) maintenance personnel, with a cost favorable impact on related hardware and software developers.

1.2.2 Assure the broadest application of programming language standards for description of end item (subassembly, assembly, and/or system) performance, including technical diagnoses and configuration management of test procedure description for support of electrical, electronic, mechanical, pneumatic, hydraulic and other systems. Primary emphasis will be directed toward application to ATE but performance and diagnostic procedure will be translatable to semi-automatic and/or manual test methods.

1.2.3 Provide impetus for industry-wide Test Procedure Language Standardization by representing the DOD in joint military-industry association activities with development, maintenance, and implementation planning for programming language standardization.

1.2.4 Provide for formal language standardization through normal standardization channels.

---

(1) "Defense Automatic Test Equipment Language Standardization Project Master Plan for Operational Performance Analysis Language System", dated 27 September 1973, by DA Defense Focal Point.

1.2.5 Recommend policies to the Assistant Secretary of Defense (Installation and Logistics) which will promote the use of a standard programming language for all (automatic, semi-automatic and/or manual) test equipment acquired in support of weapon systems.

1.3 This document describes the general characteristics and functional requirements of the language, OPAL.

2. REFERENCES

2.1 The languages described in the documents referenced below shall be used as source material in the development and specification of OPAL.

2.1.1 ATLAS

a. ARINC Specification 416-8, Abbreviated Test Language for Avionic Systems (ATLAS), Aeronautical Radio, Inc, Annapolis, MD (July 1973)

2.1.2 CTL

a. Muntz, C., et al, A Preliminary Design for a New Programming Language, Frankford Arsenal Letter Report N7000-8-73, Massachusetts Computer Associates (MCA), Waltham, MA (January 1973)

b. Loveman, D., et al, Compass Test Language (CTL) Syntax and Parser, Frankford Arsenal Letter Report N7000-9-73, MCA, Waltham, MA (January 1973)

c. Warshall, Stephen, et al, CTL-Compass Test Language, Frankford Arsenal Report R-3017, MCA, Waltham, MA (June 1974)

2.1.3 GOAL

a. Dickison, Larry R., Ground Operations Aerospace Language (GOAL) Textbook, TR-1228, NASA, John F. Kennedy Space Center, FL (16 April 1973)

b. Dickison, Larry R., Ground Operations Aerospace Language (GOAL) Syntax Diagrams Handbook, TR-1213, NASA, Kennedy Space Center, FL (16 April 1973)

c. Ground Operations Aerospace Language (GOAL) Final Report, Contract NAS10-6900, Vol 1 to 5, IBM Federal Systems Division (31 July 1973)

2.1.4 FORTRAN - ANSI STD X3.9-1966 FORTRAN

2.1.5 ALGOL

a. Naur, P., et al, Revised Report on the Algorithmic Language ALGOL-60, Communications of the ACM, Vol 6, January 1963

2.1.6 PL/1 - Basis/I-8, ECMA and ANSI Working Document, July 1972

2.1.7 Army Materiel Command, ATE Language Standardization Committee Notes and Correspondence (November 1971 through February 1973 and February 1973 through January 1974).

3. REQUIRED LANGUAGE CHARACTERISTICS

3.1 General Characteristics

3.1.1 OPAL shall be a subset of natural English, to consist of the technical terms used by test design engineers and technicians to describe the procedures required to test, diagnose, and maintain electrical, electronic, mechanical, hydraulic, pneumatic, optical, and other materiel.

3.1.2 The language shall include those constraints required to render it machinable.

3.1.3 It shall provide a means to improve communication between the end-item designer, and test systems developers, users, and maintenance personnel.

3.1.4 It shall be easy to read, to write, and to understand with minimal training.

3.1.5 All terms in the language shall be unambiguous, and fully defined, with each definition adopted from a standard technical dictionary.

3.1.6 OPAL shall be ATE independent to permit the unit-under-test (UUT) oriented description of test procedures which may be readily adapted for use with any automatic, semi-automatic, or manual test equipment.

3.1.7 The language syntax shall be formally specified in Backus Naur Form. In addition, detailed syntax diagrams shall be provided for each language construct. (See Reference 2.1.1)

3.1.8 Character Set: The character set shall consist of all upper case alpha, numeric, and special characters included in the American Standard Code for Information Interchange (ASCII), and such characters as are required to represent the International Metric System (SI) dimensional units.(2) Provisions shall be made to extend the OPAL character set to include all lower case ASCII characters when peripheral equipment capable of generating the full ASCII set is generally available.

---

(2) ISO Standard #2955, "Information Processing Representation of SI and Other Units for Systems with Limited Character Sets", Published by ANSI, 1974.

3.1.9 The language shall be expandable to allow the addition of elements and constructs as required.

3.1.10 To the maximum extent possible, the language shall be based upon existing test languages such as ATLAS (as specified in Reference 2.1.1), CTL (as specified in Reference 2.1.2), and GOAL (as specified in Reference 2.1.3) with such additions and modifications as are necessary to meet service requirements. Constructs derived from other high level languages may be included if applicable.

3.1.11 Statement Format: All statements shall be written in free format to permit the indentation and spacing of statements and statement elements for maximum readability.

3.1.12 Symbolic Names: Meaningful symbolic names shall be assigned to data (variables, parameters, messages, lists of values) and to program units. Data names (called "identifiers") are symbolic representations of memory blocks into which data will be stored. Program unit names (called "labels") are symbolic representations of pointers to parts of the program itself (statements, modules, procedures, subroutines, tasks, and blocks).

NOTE: A required part of all OPAL program documentation shall be cross-reference tables, one for all identifiers and a second for all labels. Each table shall be arranged in alphabetical order, and shall list all line numbers within which each symbolic name appears.

3.1.13 Punctuation: The rules of English grammar shall be followed as closely as possible to improve the language readability without unfavorably impacting machinability.

3.1.14 Structured Programming: OPAL shall be designed to encourage the use of the principles of structured programming. Programs written in OPAL shall be broken into labelled modules which may be compiled and debugged as self-contained entities. Breaks in the logical flow of each module shall be avoided. Copious commentary shall be included throughout each program unit to enhance clarity.

3.2 Functional Requirements

The language shall provide effective constructs for the expression of the functions described in the sections which follow.

3.2.1 Declaration of Data Type

A construct shall be provided to allow the declaration of data type, the assignment of an identifier to a declared datum or array, the designation of the dimensions and bounds of an array, and the optional specification of the initial contents of the datum or array. Data types shall include but not be limited to real, integer, and complex numbers; bit and character strings; and message texts.

3.2.2 Define Statements

3.2.2.1 Define Units

OPAL shall allow the definition of basic units followed by such derived units as are required for a given test program.

3.2.2.2 Define Procedures

A facility shall be provided to allow the definition of procedures (internal subroutines) which may be called (by name, and by the specification of actual parameters) in the main flow of a test program.

3.2.2.3 Define Task

A facility shall be provided to allow the definition of an interrupt event or condition which will trigger an interrupt, and of the task to be executed to service the interrupt.

3.2.3 Reference Statement

A reference statement shall be provided to allow the specification of common data, defined procedures, and library subroutines which are to be used within a given program module.

3.2.4 External Library

Constructs shall be provided to allow the definition and call of external library subroutines. (The addition of a new subroutine to the system library shall be accomplished in accordance with rules to be formulated for the configuration management of the OPAL System.)

3.2.5 Comments

A construct shall be provided to permit the insertion at any point in a program of comments written to clarify the meaning of a program unit. Comments shall be listed as part of the source language version of a program, and otherwise ignored by the language processors.

3.2.6 Arithmetic, Logical, and Shift Operators

The arithmetic operators for addition, subtraction, multiplication, division, and exponentiation; the logical operators AND, OR, XOR, and NOT; and the shift operators left/right shift, arithmetic shift, and rotate, shall be an integral part of the language. A construct shall be provided to assign the results of arithmetic, boolean or shift/rotate operations to a designated symbol. The precedence rules of basic algebra shall be adopted in the evaluation of arithmetic expressions. The elementary functions (all trigonometric functions, plus square root, common log, natural log, antilog, and absolute value) shall be a basic part of the language structure, and may be used by name in any arithmetic expression. Boolean operations shall be processed in accordance with the following precedence rules, whenever Boolean expressions are not parenthesized: NOT operations first, XOR operations second, AND operations third, and OR operations fourth. The use of parentheses shall be encouraged to improve readability.

Shift operations shall cause the bits of the specified variables to be shifted in the specified direction the specified number of positions. Bits shifted out of either end of a computer word are lost. All vacated bit positions are filled with zeros.

Arithmetic shift operations shall cause the sign bit (assumed to be in the most significant bit position) to remain unaltered. For a left arithmetic shift, the vacated bit positions are filled with zero. For a right arithmetic shift, the sign bit is duplicated in all vacated positions.

The rotate operation shall cause bits rotated out of either end of a computer word to reenter the opposite end. No bits are lost in this process. .

3.2.7 Block Structure

A block is defined as a sequence of related statements which may be treated as a single statement. The construct to be used to define the beginning and end of a block shall be DO...END, where within the DO...END brackets the related sequence of statements is specified.

3.2.8 Loop Control

Loop control constructs of the following forms shall be provided:

FORM 1

```
FOR I = A to B
  DO .
     .
     .
  END
```

where I is the Index, A its initial value, and B its final value. The index is incremented by 1 each time the loop is executed. The block which defines the sequence of statements to be repeated the designated number of times shall be coded within the DO...END brackets.

FORM 2

```
For I = A to B by Y
  DO
     .
     .
     .
  END
```

where the meaning of I, A, and B is the same as that specified for Form 1, and Y is designated whenever the increment on the index is a value other than 1.

FORM 3

```
FOR X units of time
  DO
     .
     .
     .
  END
```

where the block specified is executed repeatedly for the specified time duration.

FORM 4

```
WHILE condition  (See 3.2.9 a and b)

  DO  .
      .
      .
  END
```

where the specified block is executed repeatedly until the specified condition is true. A maximum time limit shall be specified as part of the condition description to preclude the occurrence of endless run-time looping.

NOTE: The convention shall be adopted to execute a loop zero times if the specified limit on the index (Forms 1 and 2), or on time (Form 3) is met, or if the condition (Form 4) is true, when a given loop is entered. All implementations of the language shall require that the run-time package check that the value of a loop index never inadvertently exceeds the specified index boundary.

FORM 5

```
FOR X = A, B,...N

  DO
     .
     .
     .
  END
```

where the block is repeated until the list of values of X specified in the FOR statement is exhausted. A through N may be constants or identifiers.

3.2.9 Conditional Statements

The construct:

```
IF condition

   THEN
    DO  ⎫
     .  ⎪
     .  ⎬  path to be followed if condition is true
     .  ⎪
    END ⎭
```

```
ELSE
  DO
   .
   .        path to be followed if condition is false
   .
  END
```

shall be provided to allow the in-line expression of the block or single statement to be executed if the specified condition is true, followed by the block or single statement to be executed if the condition is false. The 'condition' may be specified by:

a. The relational operators

EQ (is equal to)
LE (is less than or equal to)
LT (is less than)
GR (is greater than)
GE (is greater than or equal to)
NE (is not equal to)

b. The state operators

ON or OFF
TRUE or FALSE

Subroutines or procedures may be called within a given path. However, only one entry point and one exit point should be specified for each path.

3.2.10 Unconditional Transfer Statement

The construct --GO-TO statement label-- shall be provided to permit the unconditional transfer of control to the designated statement. However, this statement should be used sparingly to maximize the ease with which the logical flow of a program can be followed.

3.2.11 Interrupt Control

Constructs are required to specify:

a. The definition of events or conditions which are to trigger the execution of a given task.

b. The definition of the sequence of statements which together form the task.

c. The enabling of a task; that is to initialize it such that when a specified run-time condition exists (or event occurs) the task may be executed with minimal or no delay.

d. The disabling of a task; that is, to cause the program to ignore a given interrupt condition or event which was previously enabled.

e. The assignment of priorities to each interrupt such that should two or more occur simultaneously, the run-time operating system can "know" which to service first, and in what order to service the others.

3.2.12 Perform

A construct shall be provided to call for the execution of a defined procedure, or program module, or subroutine or task. Actual parameters to be manipulated when the program element is performed shall be listed in the perform statement, to replace the formal parameters (if any) specified in the definition of the program element.

3.2.13 Apply Statement

A construct shall be provided to describe (1) the application of stimuli and/or loads to a UUT, (2) the required characteristics of the elements to be applied, and (3) the test point(s) or connector(s) on the UUT to which the stimuli and/or loads will be connected.

3.2.14 Remove Statement

A construct shall be provided to describe the removal of stimuli and/or loads from designated test point(s) of the unit under test.

3.2.15 Measure Statement

A construct shall be provided to describe (1) the measurement of the physical characteristics of the UUT, (2) the symbolic name of the measurement read, and (3) the UUT test point(s) at which the measurement is to be made.

3.2.16 Range and Tolerance

Within both apply and measure statements, constructs shall be provided to allow the designation of:

a. The maximum, minimum, or range of values within which a measured or applied physical characteristic should lie, and the units in which the values are expressed.

b. The tolerance allowed on each applied or measured value.

c. An error exit to service the sensing of values which exceed the designated limits.

3.2.17 Compare

A construct shall be provided to allow the comparison of the value of two or more characteristics, or of two or more bit or character strings, or of the state of two or more UUT elements.

3.2.18 Verify

A construct shall be provided to allow the verification of the relation between, or state of, two or more UUT elements or characteristics.

3.2.19 Time Control Statements

All time related statements will be self contained; that is, all time measurement shall begin with the execution of the time control statement, and shall not depend on any preceeding statement.

3.2.19.1 Delay Statements

a. Delay x time units - execution of the next statement in sequence will be delayed for the specified length of time.

b. Delay Until condition - execution of the next statement in sequence will be delayed until the specified condition is true. Maximum time will be designated to provide an escape to an error exit if the condition is not true within the given maximum time.

3.2.19.2 Time Monitoring Statements

a. Start Clock - the system timer or designated external timer will be initialized to establish a time reference point.

b. Read Clock - the current value of time will be read and stored in a named location.

c. A Start Clock statement may be followed by any number of Read Clock statements. Other OPAL statements may be interspersed between the Start Clock and any Read Clock statement. Each Start Clock statement will reinitialize the designated clock. The value of time at initialization will be stored in a designated temporary location, to serve as a time reference point for subsequent Read Clock statements.

d. The Start Clock/Read Clock statements may optionally include a WHEN condition phrase within which an event or condition may be designated to trigger the Start or Read action.

3.2.19.3 Timed-Block Statements

```
Within X time units

    DO
     .
     .
     .
    END
```

All statements enclosed in the DO...END brackets shall be executed within the specified time interval. (NOTE: Implementation of the Timed-Block statements shall require that the language processor verify that the block can be executed by the run-time system within the designated time interval. If the run-time ATE cannot meet the time limit, an error message shall be generated at compile time.)

3.2.19.4 <u>Synchronization</u>

A construct shall be designed to describe the synchronization of two or more events, or of specified points on two or more waveforms.

3.2.20 <u>Event Counting</u>

A construct shall be provided to designate:

a. The description of an event.

b. An optional trigger which will initiate a count of the occurrence of the event.

c. The identifier into which the actual count will be stored.

When the count is to occur over a specified period of time, Loop Control Form 3 (paragraph 3.2.8) should be used.

3.2.21 <u>Decision Tables</u>

Decision tables will be used to present in compact, tabular form the mapping of test results onto a set of appropriate actions. The required constructs are:

a. A mechanism to allow the definition of a set of intervals within which the value of a given measurement may fall; and to name and specify the end points of each interval. The mechanism should also allow the assignment of identifiers to sets of specified bit strings.

b. A mechanism to allow the definition of decision tables composed of predefined "fault signatures" each of which represents the unique set of either measurement indicators (interval names) for each parameter measured or of bit string identifiers,which together point to a specific fault diagnosis, or to more than one possible fault. Each decision table shall represent the test designer's analysis and definition of those

faults in a given UUT, or UUT subsystem, which can be diagnosed by the measurement of the specified interactive combinations of physical properties or by the generation of specified bit strings.

c. A mechanism to allow the test designer to control the search of decision tables.

3.2.22 Digital Terms

No separate dictionary of digital terms shall be invented to describe the analog characteristics of digital signals such as waveform characteristics or timing requirements. Instead, the appropriate analog descriptors shall be used. Constructs shall be provided to:

a. Label partial words of a specified bit or character length in a designated code.

b. Pack partial words into a designated word length.

c. Mask and unpack partial words.

d. Initiate the input of tables of bit and/or character strings; designate the required response to each input pattern, and the action to be taken if the actual response pattern deviates from the required response. Indicate whether the table is to be processed under software or hardware control.

3.2.23 Insertion of Non-OPAL Program Segments

A construct shall be provided to allow the insertion of program segments written in a non-OPAL language. The "Leave/Resume" brackets used in ATLAS for this purpose may be adopted.

3.2.24 Input/Output Control

Constructs shall be provided to allow the programming of communication between the program-controlled ATE hardware and the operator. The constructs shall include mechanisms to initiate displays, output message texts, output visual or auditory signals, and accept "push-button" or other input from the operator.

3.2.24.1 Output

A single output construct shall be provided which shall include:

a. The identifier(s) or actual string(s) of information to be output.

b. The name of the kind of peripheral device to which the output is to be sent.

c. An optional reference to a format statement in which the desired format has been specified.

3.2.24.2 Input

A single input construct shall be provided which shall include:

a. The specification by identifier of the location(s) into which the input is to be stored.

b. The peripheral device from which the information is to be obtained (or the designation of the operator as the source of input).

c. An optional reference to a format statement.

3.2.25 Format Statement

A construct shall be provided within which a full description of the layout of data or text to be input or output may be specified.

APPENDIX I

Derivation of OPAL Concepts and Constructs

In the following list, the derivation of the concepts and/or constructs included in the functional description of OPAL are specified.

Item and Paragraph Number in OPAL Functional Description	Language from which Derived
1. DATELS Project Objectives	
1.2.1 Standard to improve Communication	ATLAS, CTL
1.2.2 For ATE, Semi-ATE and Manual Equipment	ATLAS (extended to consider non-avionic UUTs), CTL
2. General Characteristics of OPAL	
3.1.1 Natural English subset	ATLAS (Modified and extended), GOAL, CTL, AMC Committee
3.1.2 Machinable	
3.1.3 Improve Communications	
3.1.4 Easy to Read, Write, Understand	
3.1.5 Definitions from Technical Standard Dictionaries	AMC Committee
3.1.6 ATE Independent	ATLAS, GOAL, CTL, AMC Committee
3.1.7 (1) Formal Definition in BNF	ALGOL, CTL, GOAL, PL/1
(2) Syntax Diagrams	ATLAS, GOAL, AMC Committee
3.1.9 Expandable	CTL, AMC Committee
3.1.11 Statement Format	PL/1, CTL, GOAL
3.1.12 (1) Labels and Identifiers	ALGOL, PL/1, CTL
(2) Program documentation note	AMC Committee

Item and Paragraph Number in OPAL Functional Description	Language from which Derived
3.1.14 Structured Programming	CTL, AMC Committee, PL/1
3. Functional Characteristics of OPAL	
3.2.1 Declaration of data types and dimensions of arrays	FORTRAN (Limited), ALGOL, PL/1, ATLAS, CTL, GOAL, AMC Committee
3.2.2.1 Define Units	CTL
3.2.2.2 Define Procedures	ALGOL, ATLAS, CTL, AMC Committee, PL/1 (called a subroutine in GOAL)
3.2.2.3 Define Task	AMC Committee, PL/1
3.2.3 Reference Statement	PL/1, CTL
3.2.4 External Library	FORTRAN, ALGOL, PL/1, CTL, AMC Committee
3.2.5 Embedded Comments	PL/1, CTL, AMC Committee
3.2.6 (1) Arithmetic and Logical Operators	FORTRAN, PL/1, ALGOL, ATLAS, GOAL, AMC Committee, CTL
(2) Shift/Rotate Operators	ATLAS, AMC Committee
3.2.7 Block Structure	PL/1, CTL, AMC Committee, ALGOL (modified)
3.2.8 Loop Control	
(1) Forms 1, 2, and 5	FORTRAN (modified), ALGOL (modified)
(2) Forms 1, 2, 4 , and 5	PL/1 (modified)
(3) All Forms	CTL, AMC Committee
3.2.9 Conditional Statements	ALGOL, PL/1, CTL, AMC Committee
3.2.10 Unconditional Transfer	FORTRAN, ALGOL, PL/1, CTL, ATLAS, GOAL, AMC Committee
3.2.11 Interrupt Control	ATLAS (modified), GOAL (modified), AMC Committee

APPENDIX I

Item and Paragraph Number in OPAL Functional Description	Language from which Derived
3.2.12 Perform	ATLAS, AMC Committee, GOAL (modified)
3.2.13 Apply	ATLAS,GOAL, AMC Committee, CTL
3.2.14 Remove	ATLAS, AMC Committee, CTL
3.2.15 Measure	ATLAS (modified), AMC Committee, CTL
3.2.16 Range and Tolerance	ATLAS (modified), AMC Committee
3.2.17 Compare	ATLAS, GOAL (modified), AMC Committee
3.2.18 Verify	ATLAS, AMC Committee, GOAL (modified)
3.2.19 Time Control (commentary)	CTL, AMC Committee
3.2.19.1 (a) Delay	AMC Committee, CTL, GOAL, ATLAS (modified)
(b) Delay Until	AMC Committee, CTL, GOAL
3.2.19.2 Time Monitoring	AMC Committee, CTL
3.2.19.3 Timed-Block Statement	AMC Committee, CTL
3.2.19.4 Synchronization	ATLAS (requires gross modification), AMC Committee (not finalized)
3.2.20 Event Counting	ATLAS (requires gross modification)
3.2.21 Decision Tables	CTL, ATLAS (in proposed form), AMC Committee
3.2.22 Digital Terms	ALGOL (a,b,c only), PL/1 (a,b,c only), CTL, ATLAS (modified)
3.2.23 Leave/Resume	ATLAS, AMC Committee, GOAL
3.2.24 Input/Output	{PL/1 and FORTRAN (modified), CTL, AMC Committee
3.2.25 Format	{PL/1 and FORTRAN (modified)

DISTRIBUTION

Commander
U.S. Army Materiel Command
5001 Eisenhower Avenue
Alexandria, VA 22304

1 Attn: AMCRD-GP, D. Weidhuner
1 Attn: AMCRD-GP, J. Fitzgerald
1 Attn: AMCMA-T, CPT K. Allred
1 Attn: AMCDL, J. A. Bender
1 Attn: AMCQA-P, J. Vogt

Commander
U.S. Army Armament Command
Attn: AMCRDS, Mr. A. Gausz
Rock Island, IL 61201

Commander
U.S. Army Maintenance Management Center
Lexington, KY 40507

1 Attn: Mr. Charles Johnson
1 Attn: Mr. P. Smith

Commander
U.S. Army Armament Command
Attn: AMSAR-I-I, L. Goldsmith
Dover, NJ 07801

Commander
U.S. Army Troop Support Command
4300 Goodfellow Boulevard
St. Louis, MO 63120

1 Attn: AMSME-ME, J. Perry
1 Attn: AMSME-ME, R. Branstetter
1 Attn: AMSME-ME, J. Musarra
1 Attn: AMSME-ME, K. Yoast
1 Attn: AMSME-ME, T. Kilgore

Commander
U.S. Army Computer Systems Command
Attn: Liaison Office
Fort Belvoir, VA 22060

Commander
U.S. Army Aviation Systems Command
P. O. Box 209
St. Louis, MO 63288

1 Attn: AMSAV-FEF, R. Heidenreich
1 Attn: AMSAV-FEF, R. E. Tierce
1 Attn: AMSAV-FEF, CWO D. Johnston

Commander
U.S. Army Electronics Command
Fort Monmouth, NJ 07703

1 Attn: AMSEL-TL-MI, F. M. Crawley
1 Attn: AMSEL-TL-MI, L. Heiden
1 Attn: AMSEL-NL-P, Dr. E. Lieblein
1 Attn: AMSEL-NL-G, G. Brown

Commander
U.S. Army Safeguard Logistics Command
Attn: AMSSEL-MEE, C. Lienhard
P. O. Box 2400
Huntsville, AL 35804

DA-ODCSLOG
Attn: DALO-SMM-E, W. Nichols
Washington, DC 20310

Commander
Technical Library, Bldg 313
Aberdeen, MD 21005

Commander
U.S. Army Metrology and Calibration
Center
Redstone Arsenal
Huntsville, AL 35809

Commander
U.S. Army Mobility Research and
Development Center
Attn: SMEFB-RDE-HM, J. McLean
Fort Belvoir, VA 22060

Commandant of the Marine Corps
Code CSY-12
HQ, U.S. Marine Corps
Attn: C.F. Day
Washington, DC 20380

Commander
Naval Material Command
Washington, DC 20360

1 Attn: MAT 034T, D. Martin
1 Attn: MAT 0413, J. Ratzkin

Commander
Naval Electronics Laboratory Center
271 Catalina Boulevard
Attn: Irvin Rose, Code 4050 (2)
San Diego, CA 92152

Commander
Naval Electronic Systems Command
Code 4804
Washington, DC 20360
Attn: . G. Margulies

Naval Air Engineering Center
Philadelphia, PA 19112

1 Attn: SE-316-A, F. Liguori
1 Attn: SE-316-A, H. Ousey

HQ, SAAMA-MMDO
Kelly Air Force Base
Attn: B. Williams
San Antonio, TX 78241

HQ, SAAMA/XRS
Kelly Air Force Base
Attn: F. Dickson
San Antonio, TX 78241

Defense Documentation Center (12)
Cameron Station
Alexandria, VA

LT Kent Schlussel
AFSC/ASD
Wright-Patterson Air Force Base
Dayton, OH 45433

Commander
Harry Diamond Laboratories
Attn: AMXDO-TIB
Washington, DC 20438

Commander
U.S. Army Missile Command
Huntsville, AL 35809

1 Attn: AMSMI-RLE, G. Hubbard
1 Attn: AMSMI-RLE, D. Thurman

Commander
USASTRATCOM
Fort Huachuca, AZ 85613

1 Attn: SCC-LOG-M, CW2 A. L. Simmons
1 Attn: SCC-LOG-M, CW3 R. L. Reed

Commander
Army Satellite Communications Command
Attn: AMCPM-SC-8B, E. F. Wnek
Fort Monmouth, NJ 07703

Commander
U.S. Army Tank-Automotive Command
Warren, MI 48090

1 Attn: AMSTA-MSE, T. Reeves
1 Attn: AMSTA-RGD, D. Ancona

Commander
U.S. Army Test and Evaluation Command
Attn: AMSTE-RA
Aberdeen Proving Ground, MD 21005

Army Security Agency
Attn: IALOG/RME, A. Rose
Arlington, VA 22212